Towards a Composition Model Problem Based on IEC61850

Otto Preiss[1]
Department of Information Technologies
ABB Corporate Research Ltd
5405 Baden -Dättwil, Switzerland
+41 56 486 80 69
otto.preiss@ch.abb.com

Alain Wegmann
Department of Computer Science
Swiss Federal Institute of Technology
1015 Lausanne, Switzerland
+41 (21) 693 43 81
alain.wegmann@epfl.ch

ABSTRACT
In order to define an interesting model problem for research work around the assembly of software components, this paper introduces the substation automation domain. Special emphasis is put on those aspects of the application area which make its employment as model problem provider attractive. Firstly, the composition is required to meet several quality requirements besides functionality and must therefore allow for the prediction of these qualities before the final system is assembled and is undergoing a potentially costly factory and/or onsite acceptance test. Secondly, the currently known functionality of the application domain is sufficiently modeled, so that the mapping to components and their quality requirements is straightforward. Thirdly, the upcoming IEC61850 standard, which provides the basis for the functional models as well as the quality requirements, is publicly accessible. It also contains representative sample systems of various complexities.

Keywords
Software components, component assembly, quality attributes, substation automation, IEC61850

1 INTRODUCTION

The world of automation systems is traditionally rather standardized, at least with respect to hardware, hardware interfaces, field buses, etc. However, the standards are rather low level and usually application independent. For example, field bus standards define the communications profile but not the semantics of the data. The upcoming IEC61850 standard "Communications Networks and Systems in Substations" [1] goes one step further. While being a communications standard for Substation Automation (SA), emphasis was put on domain modeling, and the definition of the application data exchange. Among others, the result is a model of the SA functionality where system operations are modeled based on collaborations of atomic functional units. Although not mentioned or promoted in the standard, this lends itself very well to a rather straightforward mapping to a software component structure. The basic idea is to realize such functional units as software components and assemble customer specific SA applications through a predictable design- and later runtime component composition approach.

2 APPLICATION DOMAIN

Electric energy generation, transmission, and distribution are a fundamental need of our society. The basic requirements from the user perspective are to obtain high quality power with no interruptions at a low price. Networks (power grids) of different topologies are responsible to transport energy over short or long distances and finally distribute it to end-consumers (such as households and companies). The nodes in such a network are called substations and take over the voltage transformation and/or the routing of energy flow by means of the installed switchgear. Substations may be manned or unmanned depending on the importance of the station and also on its degree of automation. Power grids are controlled through Supervisory Control and Data Acquisition (SCADA) and Energy Management Systems (EMS) applications. They support energy trading and dispatching through remote operator access to substation equipment. Substations are controlled by Substation Automation Systems (SAS). Since unplanned network outages can be disastrous an SAS is composed of all the electronic equipment that is needed to continuously control, monitor, and protect the network. This covers all the high voltage equipment outside the substation (overhead lines, cables, etc.) as well as those inside the substation (transformers, circuit breakers, etc.). In addition, SAS provide the aforementioned remote access facilities for SCADA and EMS applications.

[1] Corresponding author

An SAS can be classified as a distributed soft real-time system. It is usually composed of 20…100 Intelligent Electronic Devices (IED) which are connected by a communications network with possibly routers and gateways. Any device that can run executable program code and provides a communications interface would classify as an IED, which also includes a workstation PC and others. While some real-time critical functions are executed more or less autonomously on a single IED, other functions are realized in distributed form over many IEDs.

The IEC61850 Standard

The standard consists of a set of documents, divided into 10 parts, which circle around four major issues: a *functional model* of the SA application domain (Part 5), a *data model* for SAS, *communications protocols* and their services (Part 7 and Parts 8/9), and an XML based *substation configuration language* (SCL, Part 6).

The relevant part for the component composition idea and thus this paper is Part 5 and partly Part 7. The former contains the functional decomposition of the application domain as well as the quality requirements on the systems in general (with references to [2]). A particular emphasis is put on the performance requirements on the data exchange between the functional units. The interesting sections of Part 7 are those describing the semantics of the exchanged application data.

The Relevant IEC61850 Concepts

This section introduces some of the terminology and a few basic modeling concepts, which are used in the standard and needed to comprehend their relevance for component-based software design approach. More details can be found in the standard itself [1] and a summary of the major abstractions and their relationships represented as UML models in [3].

The functionality of an SAS is described by the set of functions that operate in the substation environment, as seen in the left branch of Figure 2-1[2]. Functions are realized by primitive, atomic functional building blocks, the so-called Logical Nodes (LN). The physics of an SAS is modeled as a distributed system of interconnected devices (see right branch of Figure 2-1). A certain SAS function is considered distributed if it is realized through the deployment of its collaborating LNs to two or more IEDs (see also Figure 2-2). The association between LNs is the Logical Connection (LC). In architectural terms, it models the connector between LNs, i.e. it represents a communication association. The data exchange over LCs is modeled and described by the concept of Data Objects (DO). DOs define the semantics of the exchanged data in the form of domain specific, abstract data types. In addition, abstract DO wrappers are defined, which classify the performance requirements on the LN data exchange and thus provide the basis for the calculation of communications performance.

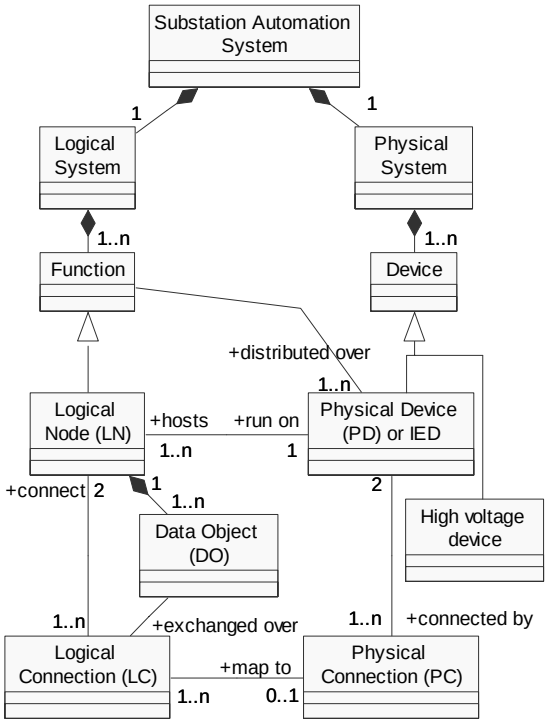

Figure 2-1: Conceptual model of the relevant abstractions of IEC61850

Figure 2-2 presents an instance diagram with a fictitious usage of the concepts shown in Figure 2-1.

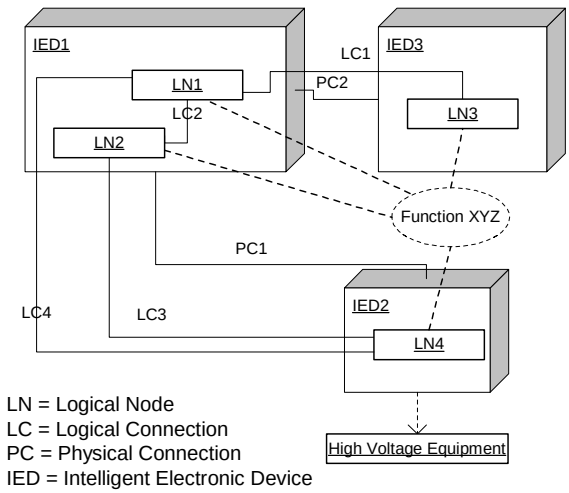

Figure 2-2: An SA function is realized by a set of collaborating LNs distributed over IEDs

Theoretically, the logical structure has no predefined allocation to the physical structure, i.e. LNs may be

[2] The role annotations of associations are used to support the reading of the conceptual diagram, e.g., a Function is "+distributed over" 1..n Physical Devices.

distributed in any way and thus are not bound to certain types of IEDs. Practically, the behavior related quality requirements together with the limited physical resources (network bandwidth, processing power) constrain the number of feasible mapping scenarios.

3 SA COMPONENTS AND SYSTEM PROPERTIES

As defined by Bass et al. [4] for software architecture, quality attributes fall into two classes. The first one refers to attributes that are discernable at system execution time (performance, etc.) and the second one to those that are not discernable at execution time, but usually over the product life cycle (reusability, etc.). In [5] we provide a more exhaustive list of the different quality attributes and their relationship to software components. However, in the context of SA systems and the model problem idea we concentrate on the first category because it is of primary concern to the end users and usually specified in requirements' documents. "Qualities of behavior" belong to the behavioral specification of systems. Hence, from the software engineering viewpoint they pertain to use cases and must be contractually satisfied by the realizations of use cases (as depicted in the collaboration pattern of Figure 3-1). In our case the realizations are collaborations of application components.

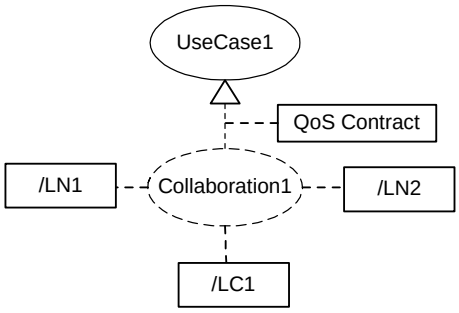

Figure 3-1: A QoS contract as an association class for collaborations that represent use case realizations

In the sequel, we refer to the set of quality attributes, which are discernable at run time and can be tied to a particular use case, as quality of service (QoS).

Relevant Quality Attributes

The generally relevant and specified QoS attributes [2] [1] for SA systems are the following:

(a) Performance; responsiveness, throughput, accuracy, timeliness, integrity, consistency, footprint, scalability
(b) Dependability; availability, reliability, degradability, fault tolerance, maintainability
(c) Security; measures against illegitimate use, denial of service attacks
(d) Safety (not discussed here)
(e) Usability (not discussed here)

The example in the next section is provided to give both, an intuition of how application compositions could look like and what kind of quality requirements is stated at use case level. It does not discuss any details of the semantics of the components or the nature of the data exchange. The fundamental idea underlying the rest of this paper is to realize LNs as software components (as depicted in Figure 3-2) and assemble application-level software based on a component instance composition approach.

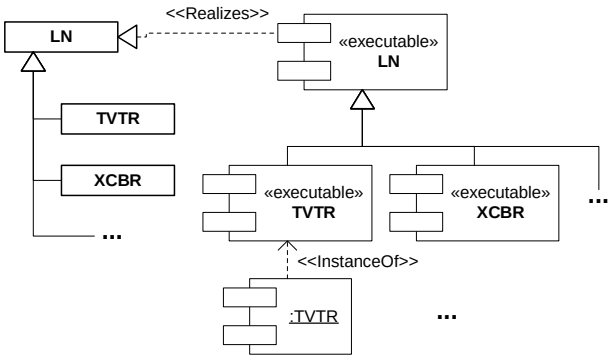

Figure 3-2: LNs realized as software components

A Sample Composition and its Requirements

Figure 3-3 shows a diagram of a simple composition that realizes three use cases. The schematically depicted high voltage equipment (circuit breaker Q0, current and voltage measurement transformers T1 and T2)[3] is the relevant part of a substation to energize the overhead line to city X. The SAS equipment consists of two nodes: IED1 and IED2. IED1, an embedded controller, is connected to IED2, a PC, via a LAN-type physical connection. IED1 and IED2 host seven and two instances of components, respectively. The components XCBR, TCTR, and TVTR realize proxies of the physical equipment (thus, their dependency association to the respective switchgear). The other components realize the specific business logic and IHMI the visualization. The use cases with their quality constraints are briefly defined.

1. Use case: *Display of measurements*. The operational values of the overhead line, actual current in Amperes and actual voltage in Volts, must be display for operators. This use case is realized through the following components: TCTR - MMXU - IHMI for current, TVTRR - MMXU - IHMI for voltage. Special quality requirements apply:
   - Age of displayed value: < 3sec
   - Accuracy of displayed value: +/- 2%
   - Availability: 99, 0%
   - Ripple display damping: 0.2 Hz
2. Use case: *Over-current protection* for the overhead line. If the measured current exceeds an adjustable limit the corresponding circuit breaker (Q0) needs to be opened. This is realized by the collaboration of TCTR - PIOC - CCBC - XCBR, whereby POIC encapsulates the

[3] A circuit breaker is a high voltage switch, voltage measurement transformers are high voltage sensors

measurement detection logic and CCBC the control logic for the circuit breaker activation. Special quality requirements apply as follows:

- Operation time: < 100ms
- Availability of functionality: 99.95%
- Reliability: MTBF[4] > 4000h
- Fault tolerance: autonomous from other components and equipment (e.g. IED2 or LAN)
- False tripping: 1/10'000, also in presence of non-plausible TCTR data.
- Data integrity: residual error probability 10^{-6} at 10^{-4} bit error probability

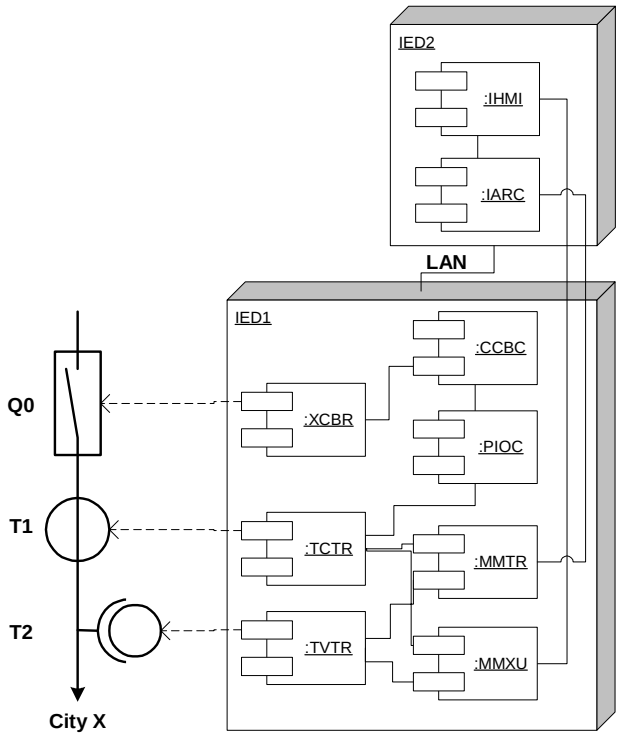

Figure 3-3: Simple protection and measurement example

3. Use Case: *Revenue Metering* (billing). The line measurements must be used for metering purpose (e.g. to do billing for city X). Metered values shall continuously be stored in one-minute intervals, but displayed to the operator on request. The use case is realized by the collaboration of TCTR - MMTR - IARC - IHMI. Special quality requirements apply as follows:
   - Measurement accuracy: 99.999%
   - Availability of functionality: 99.95%
   - Minute accuracy: +/- 1 sec
   - Degradability: recovery from missing measurements
   - Fault tolerance: must function without loss of data for 48h if IED2 or LAN fails
   - Redundant storage of data for up to 12 month without manual intervention
   - Data integrity and security: Residual error probability $< 10^{-6}$ in the presence of security attacks, link congestion and data transmission errors

Variations of the example are conceivable. For instance, IED1 could directly connect to the Internet and IED2 be a Web-enabled cellular phone for service personnel.

4 CURRENT DESIGN APPROACH

Our current R&D approach to designing such systems is to model LNs as design-time software components and apply an instance composition approach. The basis is an architectural description of the physical and logical configuration of an SAS as well as of the substation high voltage configuration. The descriptions are modeled and implemented as attributed connected graphs. Graph nodes represent instances of typed components or physical devices depending on whether the logical or physical structure is modeled. Graph edges represent typed connectors either between physical devices or software components. The attributes, which can be of any type, are used to hold domain-related information, which in turn is used by a composition rule engine to dynamically check for reasonable compositions. One mandatory attribute of a graph node holds its connection points and their type information.

The graph implementation is generic and entirely realized in Microsoft COM. This enables its independent reuse for all relevant architectural descriptions and allows any COM-compliant client (e.g. the rule engine) to access the graph and reason about the composition. The current checking algorithm is applicable to the hardware configuration as well as the component composition. The algorithm is essentially based on the following checks:

(a) Node connection point type to connector compatibility.
(b) Node connection point type to node connection point type compatibility (optional).
(c) Node to node compatibility.
(d) Fan out; Number of supported connectors per connection point.
(e) Envisioned for the future: Resource and quality attribute related checks.

5 DISCUSSION

We conclude this paper with a brief summary why we think the presented problem area is both rich enough to explore different aspects of quality attributes and easy enough to prevent a large learning effort to just relate the application domain to component based software:

(a) Since the IEC61850 provides comprehensive and agreed upon models for the application domain, research can immediately concentrate on the component assembly

[4] Mean Time Between Failures (MTBF) or Mean Time To Failure (MTTF)

related issues at application level.

(b) SA systems provide a comprehendible and controlled environment through the relative small number of application components (ca. 100). Since the interface semantics is specified at an abstract level only (and not through concrete APIs and technologies), explorative work with different types of APIs, structures, and notations, such as IDL, SOAP contract language, or Reusable Asset Specification (RAS) [6], are conceivable.

(c) Exploration with quality attribute related extensions to IEC61850's substation configuration language to describe extra-functional properties of the system and its components is conceivable.

(d) The application domain lends itself very well to investigating isolated problems of local area distributed applications with strict real-time constraints (protection), with wide area relaxed real-time requirements (SCADA/EMS), and wide area no real-time requirements (asset management or Internet-based monitoring).

The foremost reason, however, that holds against SA systems as a feasible model problem area is that there are no known manufacturers yet, who realize their SA systems in a pure component-based way, i.e., where components map to LNs, which, among others, are deployable as independent binary units. Should a component-based approach not turn out to be economical, SA systems might never be built in the way presented in this paper. This would leave us with an abstract, albeit concrete, problem, while preventing a possible proof of concept in real, productive systems.

Outlook

The pressure for vertical integration, i.e. the provision of process and automation system information into back office applications, helps to bridge the gap from the industrial automation domain to mainstream business IT. This would enrich the component assembly problem with business system specifics. Moreover, the vertical integration trend together with the increased focus on Web-services[5] could transform the discipline of a predictable component assembly process into a predictable Internet service assembly process.

ACKNOWLEDGEMENTS

We wish to thank Wolfgang Wimmer, who is a member of the IEC-TC57 WG11, for his verification of this article with respect to IEC61850 correctness.

REFERENCES

1. IEC-TC57-WG10/11/12, "IEC61850: Communications Networks and Systems in Substations," International Electrotechnical Commission, Draft IEC61850, 1999.
2. IEC870-4, "Telecontrol equipment and systems - Part4: Performance requirements," International Electrotechnical Commission, Geneva, International Standard March 1990.
3. O. Preiss, "The Major Abstractions and Models of IEC 61850," ABB Corporate Research, Baden, Technical Report TN-NST 00-3063, Sep. 14 2000.
4. L. Bass, P. Clements, and R. Kazman, *Software Architecture in Practice*, 6 ed: Addison-Wesley, 1999.
5. O. Preiss and A. Wegmann, "On Quality Attribute Based Software Engineering," Draft for submission to CBSE Workshop at 27th Euromicro 2001, Warsaw, Poland, 2001.
6. "RAS - Reusable Asset Specification," Rational Software Corporation and Catapulse, Inc., Draft Recommendation Nov. 03 2000.

[5] Web-pages that provide a programmatic interface for Internet clients